软件架构与设计模式

姓名 | 彭嘉琦

学号 | 1452697

选题：题目二

01 设计模式简介

02 三种设计模式

CONTENTS

03 观察者模式

04 进展与安排

PART ONE

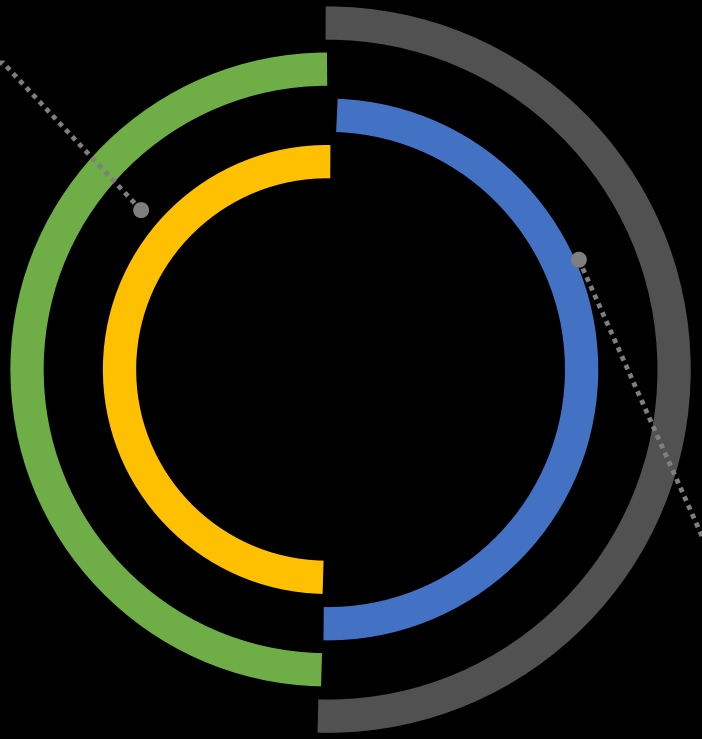
设计模式简介

01

设计模式简介

是什么

软件开发人员在软件开发过程中面临的一般问题的解决方案



使用设计模式是为了重用代码、让代码更容易被他人理解、保证代码可靠性。

为什么用

PART TWO

三种设计模式

02

三种设计模式

Composite

将对象组合成树形结构以表示“部分-整体”的层次结构，组合模式使得用户对单个对象和组合对象的使用具有一致性。

Command

将一个请求封装为一个对象，从而使我们可用不同的请求对客户进行参数化；对请求排队或者记录请求日志，以及支持可撤销的操作。

Observer

定义对象间的一种一对多依赖关系，使得每当一个对象状态发生改变时，其相关依赖对象皆得到通知并被自动更新

PART THREE

观察者模式

03

观察者模式

Observable

virtual void addObserver(Observer* observer);
virtual bool deleteObserver(Observer* observer);
virtual void notifyObservers();
void setChanged();
void clearChanged();

-list<Observer*> observers;
-bool changed;

Observer

virtual void update();

Heater

void change();

Listener

void update();

-string name;

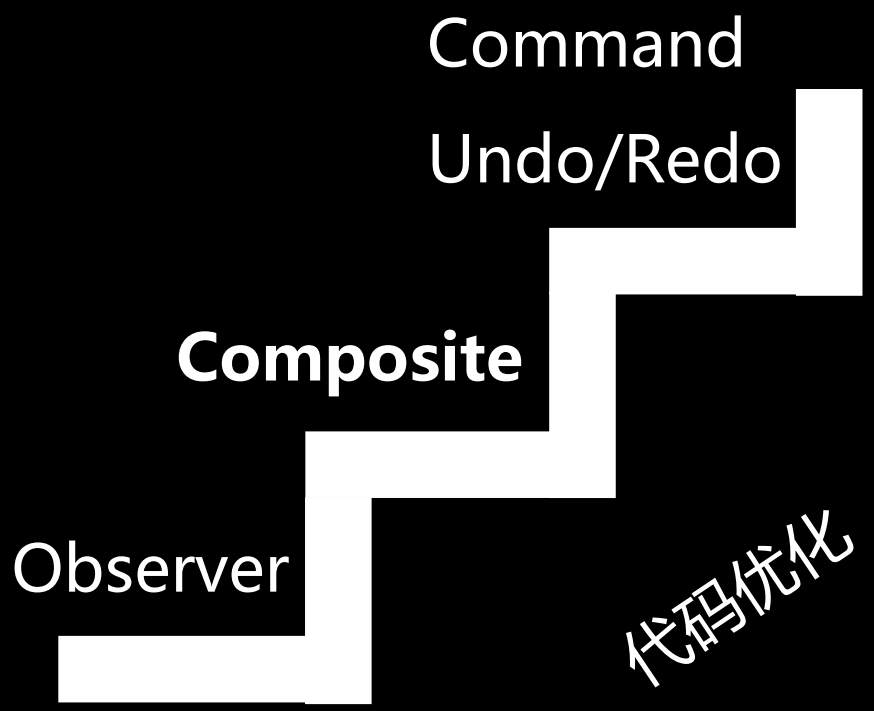
```
#include "Heater.h"
#include "Listener.h"
#include <iostream>
using namespace std;

int main(){

    Heater* heater = new Heater();

    Listener* John = new Listener("John");
    heater->addObserver(John);
    heater->addObserver(new Listener("xiao hong"));
    heater->addObserver(John);
    heater->change();

    heater->deleteObserver(John);
    heater->change();
    return 0;
}

//output
Time Up!
John is going to fetch water.
xiao hong is going to fetch water.
Time Up!
xiao hong is going to fetch water.
```

PART FOUR

进展与安排

04

进展与安排



THANK YOU!

姓名 | 彭嘉琦

学号 | 1452697

选题：题目二